IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Appellant(s):	Elmootazbellah N. Elnozahy et al.		
Assignee:	International Business Machines Corp.		
Title:	HARDWARE SUPPORT FOR SUPERPAGE COALESCING		
Serial No.:	10/713,733	Filing Date:	November 13, 2003
Examiner:	A. Savla	Group Art Unit:	2185
Docket No.:	AUS920030760US1		

Austin, Texas
March 5, 2008

COMMISSIONER FOR PATENTS
Mail Stop Appeal Brief—Patents
P.O. Box 1450
Alexandria, VA 22313-1450
—filed electronically via EFS-Web—

APPEAL BRIEF UNDER 37 C.F.R. §41.37

Dear Sir:

This appeal brief is submitted pursuant to the notice of appeal filed January 8, 2008. Please charge the fee of $510.00 due under 37 C.F.R. §41.20(b)(2) to deposit account number 09-0447. No extension of time is believed to be required in filing this brief; however, in the event any extension of time is required, please consider that extension requested and please charge any associated fee, and any additional required fees, to deposit account number 09-0447.

REAL PARTY IN INTEREST

The real party in interest to this appeal is International Business Machines Corporation, the assignee of record.

RELATED APPEALS AND INTERFERENCES

There are no other pending appeals, interferences or judicial proceedings known to Appellants or their legal representative which may be related to, directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

STATUS OF CLAIMS

Claims 1-20 are currently pending in this application. Claims 21-23 have been canceled. Claims 1-20 have been rejected. The rejection of Claims 1-20 is being appealed. These claims are set forth in the Claims Appendix attached hereto.

STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the final rejection that is the subject of this appeal.

SUMMARY OF CLAIMED SUBJECT MATTER

There are three independent claims pending which have been finally rejected, Claims 1, 7 and 14. Annotated copies of these claims are set out below indicating in bold face between braces { } where the claim feature is described in the specification by reference to page and line numbers, and to the drawings:

Claim 1

A method of assigning virtual memory to physical memory in a data processing system, comprising the steps of:

allocating a set of physical memory pages of the data processing system for a new virtual superpage mapping **{p. 14, l. 2-7; Fig. 6, ref. numeral 92}**;

instructing a memory controller **{p. 10, l. 21-27; Fig. 3, ref. numeral 42}** of the data processing system **{p. 11, l. 18-27; Fig. 4, ref. numeral 52}** to move a plurality of virtual memory pages corresponding to an old page mapping to the set of physical memory pages corresponding to the new virtual superpage mapping **{p. 14, l. 6-8 ; Fig. 6, ref. numeral 94}**; and

accessing at least one of the virtual memory pages using the new virtual superpage mapping **{p. 14, l. 8-9; Fig. 6, ref. numeral 96}** while the memory controller is copying old physical memory pages to new physical memory pages **{p. 14, l. 13-20; Fig. 6, ref. numeral 104}**.

Claim 7

A memory controller comprising:

an input for receiving remapping instructions for a virtual superpage **{p. 10, l. 21-27; Fig. 3, unnumbered horizontal arrow on left}**;

a mapping table which temporarily stores entries of old page addresses and corresponding new page addresses associated with the page remapping instructions **{p. 10, l. 27 through p. 11, l. 6; Fig. 3, ref. numeral 46}**; and

a memory access device which directs the copying of memory pages from the old page addresses to the new page addresses while handling access operations which use the new page addresses, and releases the entries in said mapping table as copying for each entry is completed **{p. 11, l. 7-14; Fig. 3, ref. numerals 48, 50}**.

Claim 14

A computer system comprising:

a processing unit **{p. 11, l. 20-23; Fig. 4, ref. numeral 56}**;

an interconnect bus connected to said processing unit **{p. 11, l. 22-25; Fig. 4, ref. numeral 58}**;

a memory array **{p. 10, l. 5; p. 12, l. 1-8; Fig. 4, ref. numeral 44}**; and

a memory controller connected to said interconnect bus and said memory array, wherein said memory controller copies memory pages from old page addresses to new page addresses according to a new virtual superpage mapping while handling access operations which use the new page addresses and while said processing unit carries out program instructions using the new page addresses **{p. 10, l. 21 through p. 11, l. 17; Fig. 4, ref. numeral 42}**.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-7, 9-17 and 19-20 were rejected under 35 U.S.C. §103(a) as being unpatentable over Appellants' admitted prior art (AAPA) in view of U.S. Patent No. 6,434,681 (Armangau). Claim 8 was rejected under §103(a) as being unpatentable over AAPA in view of Armangau and the article "Reducing TLB and Memory Overhead Using Online Superpage Promotion" (Romer). Claim 18 was rejected under §103(a) as being unpatentable over AAPA in view of Armangau and U.S. Patent No. 6,732,238 (Evans).

ARGUMENT

Appellants would respectfully submit that the following points represent errors in the final rejection dated October 10, 2007, of Claims 1-20.

A. Rejection of Claims 1-7, 9-17, and 19-20 under §103(a)

The proposed combination of AAPA and Armangau does not render Claims 1-7, 9-17 and 19-20 unpatentable because that combination does not result in a memory controller which moves virtual memory pages from an old page mapping to a new virtual superpage mapping, and can respond immediately to memory accesses using the new virtual superpage mapping even while still copying old physical memory pages to new physical memory pages corresponding to the new virtual superpage mapping. In setting forth the §103(a) rejections, the Office Action proposes many flawed analogies in comparing both AAPA and the system of Armangau to the present invention.

The first error in the Office Action is the assertion that a "processor" is the same as a "memory controller". The Office Action argues that the admitted prior art discloses the step of instructing a memory controller to move a plurality of virtual memory pages, referring to page 5, lines 16-17 of the specification which states that the operating system (OS) "uses the processor to copy data …." The Office Action explicitly asserts at the top of page 3 that "the 'processor' is

analogous to the 'memory controller.'" This assertion is incorrect. A processor of a computer system is not the same thing as a memory controller, and one skilled in the art would not consider these two different components to be analogous. The prior art computer system shown in Appellants' Figure 1 helps illustrate this point. The processor is reference numeral 22 located within the processing unit, but the memory controller is part of the system memory which is reference numeral 16. Page 10, lines 4-5, of Appellants' specification similarly explains that the memory subsystem includes a memory controller and system memory. Every computer scientist understands that the processing unit is not a part of the memory subsystem, and there is absolutely no objective basis to support such an analogy.

A processor handles generalized instructions such as arithmetic operations (floating-point or fixed-point) using various registers, as well as load/store operations, to access both memory systems and I/O devices. In contrast, a memory controller is limited to managing its associated memory array(s), and a memory controller is responsive to memory data read and memory data write instructions from a processor; indeed the hardware component that sends the instruction to the memory controller in Appellants invention is a processor. These two components have different functionalities and are clearly not interchangeable. The fact the functionally distinct elements may be fabricated on the same chip is irrelevant. The Office Action thus disregards the plain meaning of the explicitly recited term "memory controller" particularly in the claimed context of a data processing system. This interpretation of the term "memory controller" in regard to Claim 1 is also inconsistent with the interpretation given in regard to Claim 14 at page 8 of the Office Action which refers to the storage controller of Armangau as being analogous to a memory controller while distinguishing the processing unit of Armangau.

The second error in the Office Action is the implication that the prior art software procedure for superpage remapping is analogous to Appellant's claimed hardware-based method. In rejecting Claim 1 the Office Action relies on the description at page 5, lines 13-17 of Appellants' specification. According to that prior art technique, the OS (software) figures out what instructions are necessary to implement a new virtual superpage mapping, and dispatches that series of instructions to the processor which then carries out those instructions to copy pages to new locations in physical memory corresponding to the virtual superpage. The specification even notes at page 5, line 11-12, that this "solution resorts to software-directed memory

copying." Software-based page migration has many disadvantages, among them the extra overhead necessary to direct the copying which interferes with processing performance. The OS must first utilize processor resources to derive appropriate copy instructions, and then further keeps the processor busy carrying out those instructions. More significantly (and as discussed below with regard to Armangau), the new memory addresses cannot be used until after all of the copying is completed, at which time the page table entries are coalesced into the superpage. During this wait, the application continues to suffer from poor translation-lookaside buffer (TLB) behavior (see page 5, lines 18-20).

In contrast, the present invention is achieved through hardware controls without the need for dispatching such a series of instructions to the processor, thereby reducing processing overhead. The hardware support is achieved using the memory controller which directly supervises the page migration, hence the title of the present invention ("HARDWARE SUPPORT FOR SUPERPAGE COALESCING"). The memory controller receives simplified instructions which define the set of pages to be copied, and a state engine within the memory controller uses direct memory access to carry out copying (see page 10, line 25 through page 11, line 17). The OS can use the new mappings right away because the memory controller maintains a temporary mapping from the new physical pages back to the old physical pages until the new physical pages are brought up to date. As a result, the application TLB behavior improves immediately. Since the new addresses can be used even before copying is complete, the copying can further be carried out opportunistically, imparting even more operational efficiency to the memory subsystem. The means for carrying out this hardware-based control is explicitly recited in Claims 1, 7 and 14, *viz.*, the "memory controller" whose relevant hardware structure is shown in Appellants' Figure 3. Thus the distinction between software supervision of page remapping and hardware-based control is significant, and these two techniques are accordingly not analogous.

The third error in the Office Action is the assertion that Armangau accesses a virtual superpage using a new mapping while the memory controller is still copying pages. The Office Action acknowledges that the admitted prior art does not disclose accessing the virtual superpage using the new mapping while the memory controller is copying pages (pages 3, 7 of the Office Action), or using a mapping table for this function (page 5 of the Office Action); however, it is

clear that Armangau has nothing to do with page migration or the creation of a virtual superpage and never discusses these concepts by any terminology, and further does not use any supposed new page mappings while copying is still underway. Armangau is directed to a technique for backing up and restoring data in case of a storage system failure. The backup/restore process involves copying data from location A to location B then, at a later time when recovery becomes necessary, re-copying data back from location B to location A. The "snapshot copy volume" of Armangau mentioned in the Office Action is merely this backup version of data. The snapshot copy volume is not a new page mapping, that is, it does not take a set of old pages and coalesce them into a new superpage with different (more efficient) page addresses; it is just a reproduction of the data. The Office Action thus again disregards the meaning of the term "new virtual superpage mapping" which is constructed from "a plurality of virtual memory pages" as explicitly recited in Claim 1, and disregards similar terms recited in Claims 7 and 14.

Moreover, if any process is required to access data in the production volume of Armangau, it does so using the original memory locations (A) for the data, not by accessing the backup location (B). The Office Action incorrectly asserts that "the read/write access during snapshot maintenance is analogous to access operations while copying," because any such read/write access will use the original memory locations (the production set), not the ostensible new mapping which corresponds to the snapshot volume according to the Office Action interpretation (page 3). The specific relevant language of Armangau at column 2, lines 16-18, states that "the production data set is accessible to a host processor for read/write access during maintenance of the snapshot copy." Therefore, the accesses are made to "the production data set" only, and the text of Armangau never says that processor instructions use addresses of the snapshot volume. The Office Action analogy of the snapshot volume to a new virtual superpage mapping thus fails for many reasons.

Each of independent Claims 1, 7 and 14 recite memory controller-managed copying and accessing a virtual superpage using new page mappings before copying has been completed. Since the proposed combination of the admitted prior art and Armangau still fails to result in these features it accordingly cannot render the present invention unpatentable, and the Office Action fails to make a *prima facie* case of obviousness. The foregoing arguments apply equally

to dependent Claims 2-6, 8-13 and 15-20. These differences between the present invention and Armangau are also reflected in the dependent claims as explained below.

(i) Claims 3 and 9

Claims 3 and 9 specify that a read operation for an address of the new page mapping which is currently being copied is actually handled using the corresponding address of the old page mapping (described in Appellants' specification at page 14, lines 14-17). In rejecting these claims the Office Action refers solely to Armangau column 2, lines 16-18. That text of Armangau is quoted above, and says only that the production set is accessible during maintenance of the snapshot copy. There is no discussion about a read operation, or about using an old page mapping corresponding to a read address. This feature of Appellants' invention allows the read access to complete immediately without checking to see if the copying to the new virtual superpage is complete, and likewise without having to wait for the copying to complete.

(ii) Claims 6 and 16

Claims 6 and 16 specify updating an entry in a cache memory by modifying an address tag according to the new page mapping (illustrated in Appellants' Figure 5). In rejecting these claims, the Office Action refers primarily to Armangau column 10, lines 50-54. That portion of Armangau, however, says nothing about modifying a cache address tag according to a new page memory, but rather simply states the basic function of a cache system, *viz.*, the port adapter checking the cache memory first to see if a requested value resides there before accessing the other data storage devices. There is absolutely no discussion in Armangau of modifying cache address tags responsive to the construction of a new virtual superpage. This feature of Appellants' invention allows the superpage construction to complete without having to flush the contents of any affected cache memory, thereby improving the overall efficiency of the memory subsystem.

(iii) Claim 12 and 13

Claims 12 and 13 refer to the state engine within the memory access device that reads the matching old and new page addresses, and the use of a direct memory access engine that carries out the copying, controlled by the state engine (illustrated in Appellants' Figure 3). In rejecting these claims, the Office Action states that "the 'primary data storage subsystem' provides the functionality of a 'state engine,'" and that "the 'snapshot copy facility' is analogous to the 'DMA engine.'" However, the primary data storage subsystem is never described as reading paired old and new page addresses from a mapping table. In this regard the Office Action refers to Armangau (i) column 6, lines 42-48, (ii) column 8, lines 4-9, and (iii) column 13, lines 17-28. The first of these references states that: "In response to a backup command from the host 20, the primary data storage subsystem 21 accesses a primary directory 26 to find data of the physical storage unit specified by the backup command in order to initiate a process of copying the data from the primary storage 27 to the second storage 29 …." This text says nothing about sequentially reading paired old and new page addresses. The second of these references states: "The snapshot copy facility 69 includes a stored program that is executed by data processors in the primary data storage subsystem as described below with reference to Figs. 5-8. This stored program is a component of what is known as microcode for the primary data storage subsystem." This text does not describe direct-memory access. The snapshot copy facility is a program, i.e., software, but a DMA engine is hardware. As noted in Appellants' specification at page 11, lines 9-11, the DMA engine is hidden from the operating system to carry out the copying and runs in the background, that is, without interrupting program processing. In contrast, the snapshot copy facility of Armangau "is executed by data processors …." The third of these references states in relevant part that the "primary data storage subsystem responds by performing a snapshot copy, and transferring backup data from the snapshot copy to the secondary storage subsystem." This portion of Armangau again does not mention any direct-memory access functionality. The snapshot copy facility does not itself complete actual copying but rather is used to create the intermediate snapshot copy which is part of the larger backup procedure performed by the primary data storage subsystem as it transfers backup data from the snapshot copy to the secondary storage subsystem. These features of Appellants' invention allow the copying to be carried out opportunistically as noted in Appellants' specification at page 11, lines 11-14.

(iv) Claim 15

Claim 15 specifies that the processing unit includes a processor core having a TLB whose entries are updated prior to completion of page copying (described in Appellants' specification at page 14, lines 11-13). The Office Action states that the updating of the TLB is taught by Armangau at column 7, lines 49-64. This text discusses the de-allocation of primary storage locations to make them available for storing modified versions of physical storage units. The text does not refer to a processing unit, or a processing core, or a translation-lookaside buffer. The text further does not say anything about addresses being updated prior to completion of copying. As noted above, this feature allows the application TLB behavior to improve immediately.

B. Rejection of Claim 8 under §103(a)

The foregoing arguments apply to the rejection of Claim 8 since that claim depends from Claim 7, and the rejection is based primarily on the flawed combination of AAPA and Armangau. Romer is relied on only for the construction of a mapping table having paired entries. Romer does not disclose or suggest using new virtual superpage mappings while copying of the underlying memory pages is still ongoing.

C. Rejection of Claim 18 under §103(a)

The foregoing arguments also apply to the rejection of Claim 18 since that claim depends from Claim 16 and indirectly depends from Claim 14, and the rejection is based primarily on the flawed combination of AAPA and Armangau. Evans is relied on only for the cache supposedly relocating a cache entry. Evans does not disclose or suggest using new virtual superpage mappings while copying of the underlying memory pages is still ongoing.

Moreover, Evans does not actually teach relocating a cache entry based on a change of the address tag. The text of Evans at column 4, lines 27-34, is referring to unbalanced associativity size, meaning that some congruence classes may have one number of entries (e.g., 2) while other congruence classes have a different number of entries (e.g., 8); see column 6, lines

53-65 of Evans. The text of Evans at column 7, lines 48-64, is referring to the selection of an entry to store a new memory block when there is a cache "miss," that is, when the requested memory block is not currently stored in the cache. Nothing in Evans discloses or suggests moving an already cached memory block because of a change in its congruence class arising from remapping of a memory page.

CONCLUSION

The final Office Action relies on invalid comparisons to the prior art and on inconsistent interpretations of the prior art. Moreover, the proposed combination in support of the grounds of rejection still does not result in Appellants' invention as recited in each of the independent Claims 1, 7 and 14 since the process of Armangau does not use any analog of a new virtual superpage mapping in accessing pages while copying is still underway. Accordingly, the Office action fails to make out a *prima facie* case for any of the rejections.

Appellants have made a diligent effort to advance the prosecution of this application by pointing out the manifest errors in the Office Action final rejection and explaining with specificity how the claims as presented patentably define the invention over the prior art of record. In view of the arguments set forth herein, the application is believed to be in condition for allowance and Appellants respectfully request that the Board of Appeals remand this case to the Examiner with instructions to allow the claims under appeal.

Respectfully submitted,

/Jack V. Musgrove/

Jack V. Musgrove
Attorney for Appellant(s)
Reg. No. 31,986
Telephone: 512-689-6116

CLAIMS APPENDIX

The following claims are pending in this application and are involved in this appeal:

1. A method of assigning virtual memory to physical memory in a data processing system, comprising the steps of:

allocating a set of physical memory pages of the data processing system for a new virtual superpage mapping;

instructing a memory controller of the data processing system to move a plurality of virtual memory pages corresponding to an old page mapping to the set of physical memory pages corresponding to the new virtual superpage mapping; and

accessing at least one of the virtual memory pages using the new virtual superpage mapping while the memory controller is copying old physical memory pages to new physical memory pages.

2. The method of Claim 1 wherein said allocating step allocates a contiguous set of physical memory pages.

3. The method of Claim 1 wherein said accessing step includes the step of directing a read operation for an address of the new page mapping which is currently being copied to a corresponding address of an old page mapping.

4. The method of Claim 1 wherein said accessing step includes the step of directing a write operation for an address of the new page mapping which is currently being copied to both the address of the new page mapping and a corresponding address of an old page mapping.

5. The method of Claim 1 wherein said accessing step includes the step of directing a write operation for an address of the new page mapping which has not yet been copied to a corresponding address of an old page mapping.

6. The method of Claim 1, further comprising the step of updating an entry in a cache memory of the data processing system which corresponds to a memory location in the virtual memory page, by modifying an address tag of the cache entry according to the new page mapping.

7. A memory controller comprising:

an input for receiving remapping instructions for a virtual superpage;

a mapping table which temporarily stores entries of old page addresses and corresponding new page addresses associated with the page remapping instructions; and

a memory access device which directs the copying of memory pages from the old page addresses to the new page addresses while handling access operations which use the new page addresses, and releases the entries in said mapping table as copying for each entry is completed.

8. The memory controller of Claim 7 wherein said mapping table has 32 slots for receiving corresponding pairs of the old page addresses and new page addresses.

9. The memory controller of Claim 7 wherein said memory access device directs a read operation for a new page address which is currently being copied to a corresponding old page address.

10. The memory controller of Claim 7 wherein said memory access device directs a write operation for a new page address which is currently being copied to both the new page address and a corresponding old page address.

11. The memory controller of Claim 7 wherein said memory access device directs a write operation for a new page address which has not yet been copied to a corresponding old page address.

12. The memory controller of Claim 7 wherein said memory access device includes a state engine which sequentially reads the paired old and new pages addresses in said mapping table.

13. The memory controller of Claim 12 wherein said memory access device further includes a direct memory access (DMA) engine controlled by said state engine which carries out actual copying of the memory pages.

14. A computer system comprising:

a processing unit;

an interconnect bus connected to said processing unit;

a memory array; and

a memory controller connected to said interconnect bus and said memory array, wherein said memory controller copies memory pages from old page addresses to new page addresses according to a new virtual superpage mapping while handling access operations which use the new page addresses and while said processing unit carries out program instructions using the new page addresses.

15. The computer system of Claim 14 wherein:

said processing unit includes a processor core having a translation lookaside buffer (TLB) whose entries keep track of current virtual-to-physical memory address assignments; and

said TLB entries are updated for the new page addresses prior to completion of copying of the memory pages by the memory controller.

16. The computer system of Claim 14 wherein:

said processing unit has a processor core and an associated cache; and

said cache modifies an address tag of a cache entry which corresponds to a memory location in the new page addresses.

17. The computer system of Claim 16 wherein said cache modifies the address tag of the cache entry in response to a determination that the cache entry contains a valid value which is not present elsewhere in the system.

18. The computer system of Claim 16 wherein said cache further relocates the cache entry based on a changed congruence class for the modified address tag.

19. The computer system of Claim 14 wherein said memory controller includes:

a mapping table which temporarily stores entries of old page addresses and corresponding new page addresses; and

a memory access device which directs the copying of the memory pages from the old page addresses to the new page addresses and releases the entries in said mapping table as copying for each entry is completed.

20. The computer system of Claim 14 wherein said processing unit, said interconnect bus, said memory array and said memory controller are all part of a first processing cluster, and further comprising a network interface which allows said first processing cluster to communicate with a second processing cluster, said memory controller having at least one pointer for a new page address which maps to a physical memory location in said second processing cluster.

EVIDENCE APPENDIX

No evidence is being submitted with this appeal.

RELATED PROCEEDINGS APPENDIX

There are no proceedings related to this appeal.